

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR UNITED STATES PATENT

FOR

**SYSTEM, DEVICE, AND METHOD FOR CONTROLLING
ACCESS TO A MEMORY**

Inventors:

Richard J. Ely
9 Minuteman Lane
Sudbury, MA 01776

Stanley Chmielecki
22 Bulova Drive
Nashua, NH 03060

Attorney Docket No.: 2204/A59

Client Reference No.: 12024BA

Attorneys:

BROMBERG & SUNSTEIN LLP
125 Summer Street
Boston, MA 02110
(617) 443-9292

SYSTEM, DEVICE, AND METHOD FOR CONTROLLING ACCESS TO A MEMORY

5

FIELD OF THE INVENTION

The present invention relates generally to computer systems, and more particularly to memory access control in a computer system.

10

BACKGROUND OF THE INVENTION

15

In a typical computer system, it is not uncommon for multiple host applications to require access to a single memory device. Typically, memory accesses as well as memory access control functions are performed in software. This provides for relatively slow access to the memory device, and requires coordination between the various host applications.

20

SUMMARY OF THE INVENTION

25

In accordance with one aspect of the invention, a memory interface device is used to coordinate access to a memory device by a number of host applications. The memory interface device is situated between the number of host applications and the memory device. The memory interface device received memory access requests from the number of host applications, interacts with the memory device for servicing the memory access requests, and provides result/status information to the number of host applications. The memory interface device maintains a separate context for each memory access request in order to correlate each memory access request with the host application that issued the memory access request and the result/status information generated for the memory access request.

35

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing and other objects and advantages of the invention will
5 be appreciated more fully from the following further description thereof with
reference to the accompanying drawings wherein:

FIG. 1 is a block diagram showing a memory interface device used to
interface multiple host applications to a memory device in accordance with an
embodiment of the present invention;

10 FIG. 2 is a block diagram showing the relevant logic blocks of an
exemplary memory interface device in accordance with an embodiment of the
present invention;

FIG. 3 is a block diagram showing a memory interface device used to
interface a packet processor having multiple packet processing contexts to a
15 Content-Addressable Memory (CAM) in accordance with an embodiment of
the present invention;

FIG. 4 is a block diagram conceptually showing the relevant logic
blocks of the memory interface device control logic in accordance with an
embodiment of the present invention;

20 FIG. 5 is a logic flow diagram generically describing the operation of
the memory interface device control logic in accordance with an embodiment
of the present invention;

FIG. 6 is a logic flow diagram describing the operation of the
monitoring logic of the memory interface device control logic in accordance
25 with an embodiment of the present invention;

FIG. 7 is a logic flow diagram describing the operation of the
scheduling logic of the memory interface device control logic in accordance
with an embodiment of the present invention; and

30 FIG. 8 is a logic flow diagram describing the operation of the
result/status logic of the memory interface device control logic in accordance
with an embodiment of the present invention.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

In an embodiment of the present invention, a memory interface device
5 is used to coordinate accesses to the memory device by a number of host
applications. As shown in FIG. 1, the memory interface device 120 is coupled
between the host applications 110 and the memory device 130. The memory
interface device 120 receives memory access requests from the host
10 applications 110, interacts with the memory device in order to execute the
memory access requests on behalf of the host applications 110, and provides
result/status information back to the host applications 110.

In a typical embodiment of the present invention, the host applications
110 and the memory device 130 have different interface requirements. For
15 example, the host applications 110 and the memory device 130 typically have
different interface (bus) widths, interface cycles, interface signals, interface
protocols, and clocking. The memory interface device 120 interfaces with the
host applications 110 through a host interface that conforms to the host
application interface and with the memory device 130 through a memory
20 interface that conforms to the memory device interface. Thus, the memory
interface device 120 essentially converts or translates between the host
application interface and the memory device interface in order to coordinate
accesses to the memory device 130 by the host applications 110.

25 The memory interface device 120 maintains a context for each memory
access request. Each context is used to map the memory access request to the
host application 110 that issued the memory access request and to the
result/status information for the memory access request.

30 In a typical embodiment of the present invention, the memory interface
device 120 maintains a number of internal registers (referred to hereinafter as
a "context register set") for each host application 110. Each context register set
represents a context. Each context register set is used to receive memory

access requests from its corresponding host application 110 and provide result/status information to its corresponding host application 110. By maintaining a context register set for each host application 110, the memory interface device 120 is able to match each memory access request to the host application 110 that generated the memory access request and to the result/status information generated by the memory access request. However, because each host application 110 has access to one and only one context register set, each host application 110 can typically issue one and only one memory access request at a time.

In order for a host application 110 to access the memory device 130, the host application 110 issues a memory access request to the memory interface device 120. Specifically, the host application 110 writes a memory access request into its corresponding context register set by configuring its corresponding context register set accordingly. Writing to a particular register (e.g., an instruction register) typically indicates to the memory interface device 120 that the host application 110 has completed issuing its memory access request.

The memory interface device 120 monitors the context register sets in order to detect memory access requests. Typically, the memory interface device 120 monitors a particular register or register field within each context register set (e.g., an instruction register) that, when written by the respective host application 110, indicates that the host application 110 has completed issuing its memory access request. When the memory interface device 120 detects a memory access request in a particular context register set, the memory interface device 120 services the memory access request on behalf of the corresponding host application 110 by interacting with the memory device 130 on behalf of the host application 110.

The memory interface device 120 typically permits multiple memory access requests from multiple host applications 110 to be pending at any

given time. The memory interface device 120 schedules accesses to the memory device 130 in such a way that each pending memory access request is serviced and the context for each host application 110 is maintained (i.e., result/status information is correlated with its respective memory access request and memory access requests do not interfere with one another). The memory interface device 120 may execute each memory access request as an atomic operation, or, if the memory device 130 permits, may pipeline or interleave some or all of the memory access requests. The memory interface device 120 may or may not ensure that the memory access requests are serviced in the order in which they are received from the host applications 110.

When the memory interface device 120 completes a memory access request for a particular host application 110, the memory interface device 120 typically provides a signal to the host application 110. The signal indicates that the memory access request is complete and the result/status information is available.

In a typical embodiment of the present invention, the memory interface device 120 maintains a validity indicator in each context register set for indicating that the corresponding memory access request is complete and the result/status information is available. The memory interface device 120 typically clears the validity indicator in the context register set when a memory access request is received and sets the validity indicator in the context register set when the memory access request is complete and the result/status information is available. Alternatively or additionally, the memory interface device 120 may generate an interrupt or other signal when the memory access request is complete and the result/status information is available.

Thus, after issuing the memory access request, the host application 110 waits for the memory interface device 120 to signal that the memory access

request is complete and the result/status information is available before reading the result/status information from the memory interface device 120. For example, the host application 110 may suspend itself until the memory access request is complete, monitor for completion of the memory access request (e.g., by monitoring the validity indicator in its corresponding context register set), or continue with other tasks until interrupted by the memory interface device 120.

FIG. 2 shows the relevant logic blocks of an exemplary memory interface device 120. Among other things, the memory interface device 120 includes a host interface 210, a number of context register sets 220_1 - 220_N (referred to individually as a context register set 220 and collectively as the context register sets 220), control logic 230, and a memory interface 240. The memory interface device 120 communicates with the host applications 110 over the host interface 210 according to the host interface protocol. The memory interface device 120 communicates with the memory device 130 over the memory interface 240 using the memory interface protocol. The memory interface device 120 maintains a context register set 220 for each of the host applications 110 for receiving memory access requests from the host applications 110 and providing result/status information to the host applications 110. The control logic 230 monitors the context register sets 220 to detect memory access requests, services the memory access requests, interacts with the memory device 130 over the memory interface 240 in order to execute the memory access requests, and provides result/status information to the host applications 110.

In an exemplary embodiment of the present invention, the described memory interface device 120 is used in a router or other networking device to coordinate accesses to a Content-Addressable Memory (CAM) by multiple packet processing engines of a packet processor. As shown in FIG. 3, the memory interface device 120 is coupled between the packet processor 310 and the CAM 330. The CAM 330 is a memory device that is used to store certain

types of packet processing information such as routing/forwarding information. In order to process packets received over various router interfaces, the packet processing engines of the packet processor 310 search for routing/forwarding information in the CAM 330 by issuing memory access requests to the memory interface device 120. The CAM 330 essentially enables all memory locations to be searched simultaneously using a single memory access request, and typically provides a mechanism for "masking out" irrelevant bits and fields in the search (e.g., for searching based upon an address/prefix). When organized in an ordered list format, the CAM 330 typically returns the "best" match for a particular search.

The interface to the packet processor 310 is typically a 32-bit pipelined ZBT SRAM interface with fixed three-cycle operations. The interface to the CAM 330 is typically a 128-bit multicycle, wide access interface with concurrent instruction and control buses that enable memory accesses to be pipelined in order to reduce latency. Thus, with reference again to FIG. 2, the host interface 210 of the memory interface device 120 conforms to the packet processor 310 interface, and the memory interface 240 of the memory interface device 120 conforms to the CAM 330 interface.

The packet processor 310 typically includes four (4) independent packet processing engines. Each packet processing engine typically has four (4) independent packet processing contexts. Each packet processing context can independently issue memory access requests to the memory interface device 120 through a corresponding context memory set 220. Thus, each packet processing context represents one of sixteen (16) host applications 110. An exemplary memory interface device 120 therefore includes at least sixteen (16) context memory sets 220, one for each of the packet processing contexts supported by the packet processor 310. Each context memory set 220 typically includes eight 32-bit registers including, among other things, registers for providing comparand, control, and instruction information by the packet processing context and registers for providing result/status

information (including a validity indicator) by the memory interface device 120.

In order for a packet processing context to access the CAM 330, the packet processing context issues a memory access request to the memory interface device 120 by configuring its corresponding context register set 220 in the memory interface device 120 accordingly. Memory accesses include such things as search operations and various CAM maintenance operations (e.g., invalidating, moving, loading, testing). An instruction register in each context register set 220 is typically used to specify the type of memory access for a particular memory access request.

The control logic 230 monitors the context register sets 220 in order to detect memory access requests. Typically, the control logic 230 monitors the instruction register in the context register set 220 to determine when a memory access request has been stored in the context register set 220. When the instruction register is written by the respective packet processing context, indicating that the context register set 220 includes a complete memory access request, the control logic 230 services the memory access request on behalf of the corresponding packet processing context. Specifically, the control logic 230 typically clears the validity indicator in the context register set 220 and schedules the appropriate memory accesses to execute the memory access request.

The control logic 230 typically permits multiple memory access requests from multiple packet processing contexts to be pending at any given time. In order to reduce latency, the control logic 230 typically pipelines accesses to the CAM 330 over the memory interface 240 when it is able to do so. Specifically, the CAM 330 interface is a multicycle interface that enables multiple memory access operations to be performed during each memory access cycle. Under certain circumstances, it may be possible for the control logic 230 to schedule its servicing of the memory access requests in such a

way that, during a particular memory access cycle, some memory access operations relate to one memory access request while other memory access operations relate to another memory access request. Such pipelining enables the control logic 230 to begin servicing one memory access request before
5 completing another memory access request. However, because certain types of memory access requests and memory access operations can conflict, the control logic 230 monitors for memory access requests and memory access operations that conflict and executes those memory access requests and memory access cycle operations as atomic operations (i.e., without
10 pipelining). This typically involves completing any memory access operations that are in the pipeline before executing the conflicting memory access request or memory access operation.

When the control logic 230 completes a memory access request for a
15 particular packet processing context, the control logic 230 typically stores result/status information and sets the validity indicator in the corresponding context register set 220. The result/status information typically includes result data (e.g., data read from the CAM 330) as well as various status information (e.g., single match, multiple matches). The setting of the validity
20 indicator enables the packet processing context to determine that the memory access request is complete and the result/status information is available.

Thus, the control logic 230 includes logic for monitoring the context register sets 220 to detect memory access requests, logic for scheduling
25 memory access operations for servicing the memory access requests, and logic for providing result/status information. The logic for scheduling memory access operations for servicing the memory access requests includes logic for pipelining memory access operations, logic for detecting conflicting memory access requests and memory access operations, and logic for executing
30 memory access requests and memory access operations as atomic operations to prevent conflicts.

FIG. 4 conceptually shows the relevant logic blocks of the control logic 230. Among other things, the control logic 230 includes monitoring logic 410, scheduling logic 420, CAM protocol logic 430, and result/status logic 440. The monitoring logic 410 monitors the context register sets 220 to detect memory access requests and provides the memory access requests to the scheduling logic 420 for servicing. The scheduling logic 420 schedules memory access operations for the memory access requests using both pipelining (when possible) and atomic operations. The CAM protocol logic 430 generates the appropriate CAM 330 interface signals for interfacing with the CAM 330 via the memory interface 240. Coordination between the scheduling logic 420 and the result/status logic 440, as shown by the dashed line 450, enables the result/status logic 440 to correlate result/status information with its corresponding memory access request and store the result/status information in the corresponding context register set 220.

FIG. 5 shows an exemplary logic flow 500 generically describing the operation of the control logic 230. Beginning in block 502, the logic monitors the context register sets 220 for memory access requests, in block 504. The logic detects a number of memory access requests, in block 506, and services the number of memory access requests, in block 508. The logic obtains result/status information for the number of memory access requests, in block 510, correlates the result/status information with its respective memory access request, in block 512, and stores the result/status information for each memory access request in its respective context register set 220, in block 514. The logic 500 terminates in block 599.

FIG. 6 shows an exemplary logic flow 600 describing the operation of the monitoring logic 410. Beginning in block 602, the logic monitors the context register sets 220, in block 604. When the logic detects a memory access request in a context register set 220, in block 606, the logic clears the validity indicator in the context register set 220, in block 608, and provides the memory access request to the scheduling logic 420 for servicing, in block 610.

The logic 600 recycles to block 604 to monitor for subsequent memory access requests.

FIG. 7 shows an exemplary logic flow 700 describing the operation of the scheduling logic 420. Beginning in block 702, and upon receiving a memory access request from the monitoring logic 410, in block 704, determines whether execution of the memory access request will conflict with any memory access requests in the pipeline, in block 706. If execution of the memory access request will not conflict with any memory access requests in the pipeline (NO in block 708), then the logic generates the appropriate memory access operations for executing the memory access request, in block 712. If execution of the memory access request will conflict with any memory access requests in the pipeline (YES in block 708), then the logic completes all memory access operations in the pipeline, in block 710, and then generates the appropriate memory access operations for executing the memory access request, in block 712. The logic 700 recycles to block 704 to service subsequent memory access requests.

FIG. 8 shows an exemplary logic flow 800 describing the operation of the result/status logic 440. Beginning in block 802, the logic obtains result/status information for a memory access request, in block 804. The logic determines a context register set 220 for the memory access request, in block 806, stores the result/status information in the context register set 220, in block 808, and sets the validity indicator in the context register set 220, in block 810. The logic 800 recycles to block 804 to provide result/status information for subsequent memory access requests.

In the exemplary embodiments described above, the memory interface device 120 is used to interface multiple host applications 110 to the memory device 130. However, the memory interface device 120 may be used even for interfacing a single host application 110 to the memory device 130, particularly when the host application(s) 110 and the memory device 130 have

different interfaces. This is because the memory interface device 120 essentially converts or translates between the host application interface and the memory device interface in order to service the memory access requests on behalf of the host application(s). Thus, the present invention is not limited to any particular number of host applications 110.

In the exemplary embodiments described above, the memory interface device 120 maintains various internal registers for receiving the memory access requests and providing the result/status information. However, the present invention is in no way limited to such use of registers. For one example, rather than including registers, the memory interface device 120 could include a random access memory (RAM) or other type of memory through which the memory access requests are received and the result/status information is provided (e.g., through a set of descriptors). For another example, the host interface could be message-based such that the host application(s) 110 and the memory interface device 120 exchange information in the form of communication messages.

In the exemplary embodiments described above, the memory interface device 120 maintains one context register set for each host application 110, typically limiting each host application 110 to one and only one memory access request at a time. However, the present invention is in no way limited to maintaining one context register set for each host application 110. Other mechanisms may be used to permit each host application 110 to issue more than one memory access request at a time. For one example, the memory interface device 120 could maintain multiple context register sets for each host application 110, thereby permitting each host application 110 to issue multiple memory access requests. For another example, the memory interface device 120 could maintain multiple general-purpose context register sets and permit each host application 110 to issue multiple memory access requests.

In the exemplary embodiments described above, writing to an instruction register by the host application 110 signals to the memory interface device 120 that a memory access request is ready for servicing. However, the present invention is in no way limited to such use of the instruction register for signaling that the memory access device is ready for servicing. Other mechanisms may be used to signal to the memory interface device 120 that a memory access request is ready for servicing. For example, a separate register or field within a register may be used to signal to the memory interface device 120 that a memory access request is ready for servicing.

In the exemplary embodiments described above, a validity indicator is used to signal the host application 110 that the memory access request is complete and result/status information is available. However, the present invention is in no way limited to the use of a validity indicator for signaling that the memory access request is complete and result/status information is available. Other mechanisms may be used to signal that the memory access request is complete and result/status information is available. For example, the memory interface device 120 could generate an interrupt when the memory access request is complete and result/status information is available.

It should be noted that the term "router" is used herein to describe a communication device that may be used in a communication system, and should not be construed to limit the present invention to any particular communication device type. Thus, a communication device may include, without limitation, a bridge, router, bridge-router (brouter), switch, node, or other communication device.

It should also be noted that the logic flow diagrams are used herein to demonstrate various aspects of the invention, and should not be construed to limit the present invention to any particular logic flow or logic implementation. The described logic may be partitioned into different logic

blocks (e.g., programs, modules, functions, or subroutines) without changing the overall results or otherwise departing from the true scope of the invention. Often times, logic elements may be added, modified, omitted, performed in a different order, or implemented using different logic constructs (e.g., logic gates, looping primitives, conditional logic, and other logic constructs) without changing the overall results or otherwise departing from the true scope of the invention.

The present invention may be embodied in many different forms, including, but in no way limited to, programmable logic for use with a programmable logic device (e.g., a Field Programmable Gate Array (FPGA) or other PLD), discrete components, integrated circuitry (e.g., an Application Specific Integrated Circuit (ASIC)), or any other means including any combination thereof. In a typical embodiment of the present invention, the memory interface device 120 is a FPGA that is loaded with appropriate program logic to define the logic gates and registers for interfacing the packet processor 310 and the CAM 330 as described herein. The packet processor 310, memory interface device 120, and CAM 330 are components of a router or other networking device.

Hardware logic (including programmable logic for use with a programmable logic device) implementing all or part of the functionality previously described herein may be designed using traditional manual methods, or may be designed, captured, simulated, or documented electronically using various tools, such as Computer Aided Design (CAD), a hardware description language (e.g., VHDL or AHDL), or a PLD programming language (e.g., PALASM, ABEL, or CUPL).

Programmable logic may be fixed either permanently or transitorily in a tangible storage medium, such as a semiconductor memory device (e.g., a RAM, ROM, PROM, EEPROM, or Flash-Programmable RAM), a magnetic memory device (e.g., a diskette or fixed disk), an optical memory device (e.g.,

a CD-ROM), or other memory device. The programmable logic may be fixed in a signal that is transmittable to a computer using any of various communication technologies, including, but in no way limited to, analog technologies, digital technologies, optical technologies, wireless technologies, networking technologies, and internetworking technologies. The programmable logic may be distributed as a removable storage medium with accompanying printed or electronic documentation (*e.g.*, shrink wrapped software), preloaded with a computer system (*e.g.*, on system ROM or fixed disk), or distributed from a server or electronic bulletin board over the communication system (*e.g.*, the Internet or World Wide Web).

Thus, the present invention may be embodied as a memory interface device for interfacing a number of host applications to a memory device. The memory interface device includes a host interface for interfacing with the number of host applications, a memory interface for interfacing with the memory device, a number of contexts for receiving memory access requests from the number of host applications and providing result/status information to the number of host applications, and control logic for obtaining memory access requests from the number of contexts, interacting with the memory device over the memory interface for servicing the memory access requests on behalf of the number of host applications, and providing the result/status information to the number of host applications via the number of contexts. The number of host applications may include a number of packet processing contexts of a packet processor, in which case the host interface conforms to a packet processor interface. The memory device may be a content-addressable memory (CAM), in which case the memory interface conforms to a CAM interface. The number of contexts may be embodied as a number of context registers sets. In a typical embodiment, each context register set corresponds to one and only one of the number of host applications. The control logic typically includes monitoring logic, scheduling logic, memory interface logic, and result/status logic. The monitoring logic monitors the number of contexts for detecting memory access requests and provides the memory

access requests to the scheduling logic. The scheduling logic schedules memory access operations for the memory access requests. The memory interface logic generates memory interface signals for interfacing with the memory device over the memory interface. The result/status logic provides
5 result/status information to the number of host application(s). The monitoring logic may monitor a predetermined register (such as an instruction register) in each context register set to detect a memory access request. The memory interface may support pipelining of memory access operations, and the scheduling logic may pipeline a plurality of memory
10 access requests over the memory interface. The scheduling logic may determine that a plurality of memory access requests conflict, clear the pipeline, and execute at least one of the conflicting memory access requests as an atomic operation. The result/status logic correlates result/status information with its corresponding memory access request and stores the
15 result/status information for each memory access request in a corresponding context. The result/status logic may set a validity indicator in each context when the corresponding memory access is complete and the result/status information is available. The memory interface device may be embodied as a programmed programmable logic device (such as a programmed FPGA) or
20 an ASIC.

The present invention may also be embodied as program logic for programming a programmable logic device. The program logic includes host
25 interface logic for interfacing with the number of host applications, memory interface logic for interfacing with the memory device, a number of contexts for receiving memory access requests from the number of host applications and providing result/status information to the number of host applications, and control logic for obtaining memory access requests from the number of
30 contexts, interacting with the memory device using the memory interface logic for servicing the memory access requests on behalf of the number of host applications, and providing the result/status information to the number of host applications via the number of contexts. The number of host applications

may include a number of packet processing contexts of a packet processor, in which case the host interface conforms to a packet processor interface. The memory device may be a content-addressable memory (CAM), in which case the memory interface conforms to a CAM interface. The number of contexts

5 may be embodied as a number of context registers sets. In a typical embodiment, each context register set corresponds to one and only one of the number of host applications. The control logic typically includes monitoring logic, scheduling logic, memory interface logic, and result/status logic. The monitoring logic monitors the number of contexts for detecting memory

10 access requests and provides the memory access requests to the scheduling logic. The scheduling logic schedules memory access operations for the memory access requests. The memory interface logic generates memory interface signals for interfacing with the memory device over the memory interface. The result/status logic provides result/status information to the

15 number of host application(s). The monitoring logic may monitor a predetermined register (such as an instruction register) in each context register set to detect a memory access request. The memory interface may support pipelining of memory access operations, and the scheduling logic may pipeline a plurality of memory access requests over the memory

20 interface. The scheduling logic may determine that a plurality of memory access requests conflict, clear the pipeline, and execute at least one of the conflicting memory access requests as an atomic operation. The result/status logic correlates result/status information with its corresponding memory access request and stores the result/status information for each memory

25 access request in a corresponding context. The result/status logic may set a validity indicator in each context when the corresponding memory access is complete and the result/status information is available. The program logic may be embodied in a computer readable medium for loading into the programmable logic device.

30

The present invention may also be embodied as an apparatus including a number of host applications, a memory device, and a memory interface

device interposed between the host applications and the memory device for receiving memory access requests from the number of host applications, interacting with the memory device on behalf of the number of host applications for servicing the memory access requests, and providing
5 result/status information to the host applications.

The present invention may be embodied in other specific forms without departing from the true scope of the invention. The described embodiments are to be considered in all respects only as illustrative and not
10 restrictive.

11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
22